VERSION 1: raw
Category ---------------------- NCorrect --- N -------
Accuracy
alt.atheism ------------------- 26 --------- 319 -----
0.082
comp.graphics ----------------- 59 --------- 389 -----
0.152
comp.os.ms-windows.misc ------- 33 --------- 394 -----
0.084
comp.sys.ibm.pc.hardware ------ 47 --------- 392 ----- 0.12
comp.sys.mac.hardware --------- 48 --------- 385 -----
0.125
comp.windows.x ---------------- 58 --------- 392 -----
0.148
misc.forsale ------------------ 67 --------- 390 -----
0.172
rec.autos --------------------- 41 --------- 395 -----
0.104
rec.motorcycles --------------- 38 --------- 398 -----
0.095
rec.sport.baseball ------------ 30 --------- 397 -----
0.076
rec.sport.hockey -------------- 49 --------- 399 -----
0.123
sci.crypt --------------------- 41 --------- 396 -----
0.104
sci.electronics --------------- 22 --------- 393 -----
0.056
sci.med ----------------------- 30 --------- 396 -----
0.076
sci.space --------------------- 36 --------- 394 -----
0.091
soc.religion.christian -------- 47 --------- 398 -----
0.118
talk.politics.guns ------------ 32 --------- 364 -----
0.088
talk.politics.mideast --------- 26 --------- 376 -----
0.069
talk.politics.misc ------------ 14 --------- 310 -----
0.045
talk.religion.misc ------------ 238 -------- 251 -----
0.948
VERSION 2: mest
Category ---------------------- NCorrect --- N -------
Accuracy
alt.atheism ------------------- 53 --------- 319 -----
0.166
comp.graphics ----------------- 161 -------- 389 -----
0.414
comp.os.ms-windows.misc ------- 54 --------- 394 -----
0.137
comp.sys.ibm.pc.hardware ------ 126 -------- 392 -----
0.321
comp.sys.mac.hardware --------- 122 -------- 385 -----
0.317
comp.windows.x ---------------- 183 -------- 392 -----
0.467
misc.forsale ------------------ 122 -------- 390 -----
0.313
rec.autos --------------------- 150 -------- 395 ----- 0.38
rec.motorcycles --------------- 147 -------- 398 -----
0.369
rec.sport.baseball ------------ 111 -------- 397 ----- 0.28
rec.sport.hockey -------------- 146 -------- 399 -----
0.366
sci.crypt --------------------- 142 -------- 396 -----
0.359
sci.electronics --------------- 76 --------- 393 -----
0.193
sci.med ----------------------- 124 -------- 396 -----
0.313
sci.space --------------------- 117 -------- 394 -----
0.297
soc.religion.christian -------- 113 -------- 398 -----
0.284
talk.politics.guns ------------ 80 --------- 364 ----- 0.22
talk.politics.mideast --------- 76 --------- 376 -----
0.202
talk.politics.misc ------------ 36 --------- 310 -----
0.116

```
talk.religion.misc ------------ 206 -------- 251  -----
0.821

VERSION 3: tfidf
Category --------------------- NCorrect --- N -------
Accuracy
alt.atheism ------------------- 94 --------- 319  -----
0.295
comp.graphics ----------------- 205 -------- 389  -----
0.527
comp.os.ms-windows.misc ------- 186 -------- 394  -----
0.472
comp.sys.ibm.pc.hardware ------ 172 -------- 392  -----
0.439
comp.sys.mac.hardware --------- 234 -------- 385  -----
0.608
comp.windows.x ---------------- 160 -------- 392  -----
0.408
misc.forsale ------------------ 249 -------- 390  -----
0.638
rec.autos --------------------- 201 -------- 395  -----
0.509
rec.motorcycles --------------- 231 -------- 398  -----  0.58
rec.sport.baseball ------------ 173 -------- 397  -----
0.436
rec.sport.hockey -------------- 171 -------- 399  -----
0.429
sci.crypt --------------------- 164 -------- 396  -----
0.414
sci.electronics --------------- 145 -------- 393  -----
0.369
sci.med ----------------------- 147 -------- 396  -----
0.371
sci.space --------------------- 136 -------- 394  -----
0.345
soc.religion.christian -------- 116 -------- 398  -----
0.291
talk.politics.guns ------------ 103 -------- 364  -----
0.283
talk.politics.mideast --------- 70 --------- 376  -----
0.186
talk.politics.misc ------------ 53 --------- 310  -----
0.171
talk.religion.misc ------------ 223 -------- 251  -----
0.888
```

The prior data was acquired through running each of the versions in my program. In the following explanation use the pursuing legend for comprehension.

$v_j$ = specific category
$w_k$ = specific word

**<u>Version 1: Raw</u>**

The results from the raw version were very poor, with an average unweighted accuracy of 14.4%. Ignoring talk.religion.misc as an outlier, the average becomes 10.5%. The lowest accuracy for a given category was 4.5% for talk.politics.misc. The highest accuracy for a given category was 94.8% for talk.religion.misc.

**<u>Version 2: M-Est</u>**

The results from the M-Estimate version improved upon the results from the raw version. The unweighted accuracy was 31.7%. Ignoring talk.religion.misc as an outlier, the average becomes 29%. The lowest accuracy for a given category was 11.6% for talk.politics.misc. The highest accuracy for a given category was 82.1% for talk.religion.misc.

**<u>Version 3: tf-idf</u>**

The results from the tf-idf version were by far the best. The unweighted accuracy average was 43.3%. Ignoring talk.religion.misc as an outlier, the average becomes 40.9%. The lowest accuracy for a given category was 17.1% for talk.politics.misc. The highest accuracy for a given category was 88.8% for talk.religion.misc.

**Results analysis**

The reason why M-Estimate improved upon raw is due to $P(w_i \mid v_j)$ being calculated differently. The raw probability of a category given a lists of words in the test document was derived from Prior(v_j) * the product of $\frac{w_i \text{ instances count in } v_j}{\text{total word count in } v_j}$ for each w_k in test document.

The M-Est probability was derived from Prior(v_j) * the product of $\frac{w_i \text{ instances count in } v_j + 1}{\text{total word count in } v_j + \text{total count unique words seen in train.txt}}$ for each w_k in test document.

By adding one to the numerator, the program decides not to rule out the possibility of w_k appearing in test document v_j even though it did not appear in v_j for the training data. By adding the total count of unique words seen in train.txt to the denominator, the program is able to make the the probability of a new word appearing in v_j very small, but not zero because of the previously mentioned 1 addition.

However, the tf-idf version was by far the best. This is due to weighing the w_k frequency[*tf*] (of a given word position in test document) to the inverse document frequency[*idf*].

The term frequency is a measure of the probability of a word given a category, as seen in fraction form in for the analysis of raw probability. However, this tf gets multiplied by the idf before being multiplied into the total product.

The idf measures the significance of a word by computing $\log\left( \frac{\text{total number of documents in train test}}{\text{number of documents with } w_k} \right)$

Then, by multiplying the tf and idf, we get a weighted probability of w_k in v_j. For each w_k in test document v_j, the product of idf*tf gets multiplied by the previous products of w_k weighted probability. If there are no previous values, then the given value becomes the first.

By weighing the significance of a word to its probability, tfidf distinguishes itself from the other categories as the most successful at predicting categories.